DOCUMENT RESUME

ED 266 489 CS 209 595

AUTHOR Hitchcock, Richard B.
TITLE The Software Jungle: Guidelines for Evaluating English Computer Programs.
PUB DATE Apr 84
NOTE 9p.; Paper presented at the Annual Meeting of the Developmental Studies Conference (9th, Athens, GA, April 1984).
PUB TYPE Guides - Non-Classroom Use (055) -- Speeches/Conference Papers (150)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS *Computer Assisted Instruction; *Computer Software; Courseware; English Instruction; *Evaluation Criteria; *Instructional Material Evaluation; Microcomputers; Teaching Methods; *Word Processing; Writing Instruction; Writing Laboratories
IDENTIFIERS *Software Evaluation

ABSTRACT

Noting that the educational software market is expanding at a precipitous rate, but with considerable variation in quality, this paper outlines a thorough set of tangible and useful questions, based on two years of program design for English and writing lab usage. The major portion of the paper discusses criteria for evaluation in each of the following areas: (1) instructions, (2) user control, (3) screen format, (4) graphics, (5) data handling, (6) reinforcement, and (7) purpose. A sample courseware evaluation form is appended. (HTH)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION
CENTER (ERIC)
X This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

ED266489

THE SOFTWARE JUNGLE: GUIDELINES FOR EVALUATING ENGLISH COMPUTER PROGRAMS

by
Richard B. Hitchcock

Division of Developmental Studies
University of Georgia
Athens, GA 30606

Paper presented at the Ninth Annual
Developmental Studies Conference
Athens, Georgia
April, 1984

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
Richard B. Hitchcock
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

CS 209 595

In September of 1983, the National Education Association (NEA) compiled a catalog of educational software for microcomputers. This catalog listed over 7000 entries. But by January of 1984, the publication date, the NEA estimated that 2000 additional programs were available. The educational software market is clearly expanding at a precipitous rate, but with considerable variation in quality.

The necessary software evaluation is available. Some forms, like the California Library Media Consortium for Evaluation of Microcomputer Software, are good, but too often other forms will pose their evaluations in terms of imprecise questions such as "Does the program enhance retention?" Based on two years of program design for English and writing lab usage, this paper will outline a thorough set of tangible and hopefully useful questions, aimed at both programmers and lab coordinators. (See Appendix A for the evaluation form itself.)

I. INSTRUCTIONS

Since the average developmental student will probably be confronting a new medium (the computer) as well as an uncertain subject (English grammar and composition), accurate directions for both these areas are critical. Instructions should be presented clearly and legibly; presenting one sentence at time interspersed with pauses is often helpful. Graphics, highlighting, and underlining promote better visual clarity also. Examples should follow each new concept or direction. Lastly, what the exercise will teach and what the user can accomplish should be directly stated to the user before he begins.

II. USER CONTROL

If computer-aided instruction (CAI) is ever to become effective, the

control of the person at the keyboard must not be overwhelmed by the complexities of the computer. A program must run smoothly from disk insertion to completion; this fluidity encompasses several areas. First, if a program is chained to another file or program, it must go there (and back) without stoppage; a user may not know the CATALOG or RUN commands, and he should not have to. Second, a program should allow the user to type in certain information -- name, date, etc. -- and select the computer output -- number of questions, randomly or sequentially presented. (Random presentation is better for the user, but makes the instruction difficult; sequential presentation allows easier instruction, but becomes tedious to the user -- an option here is useful.) Third, the user should control movement from one screen to the next (RETURN and keystroke commands work best), and should be able to get help or leave the program at any point. Finally, a program should keep running accurately despite user errors: typing a letter instead of a number or accidentally pressing the RETURN key).

III. SCREEN FORMAT

As a user is moving from a print medium to an electronic one, the physical arrangement of material is important also. While considerable data can be stored in a computer, the screen size limits what the user can see. Most commercial disks are programmed in a forty-character, all-capital format so they can run in all computers. But forty characters (versus a more up-to-date eighty) limits the text displayed on the screen, and English programs may need capitalized and non-capitalized letters. Also, simple massing green letters on a dark screen is confusing and unreadable; formatting techniques like inverse and flashing lines, outlining, and underlining help make the screen pleasantly and clearly spaced. Finally, unlike books or

papers, where directions are always close by, a computer's directions may be many screens back. A good programming practice is to display directions on the screen or have them constantly available.

IV. GRAPHICS

Graphics fascinate program designers: the programming is creative and the visual effects are rewarding. But graphics also devour user time and great chunks of memory (fifty lines of high resolution graphics will overflow an Apple II's memory). Graphics should be useful, even essential to a program, not merely interesting and certainly not distractive. Graphics are best used in attention-getting title frames and scoring frames, to change the routine of a program. Sound is a helpful adjunct to a program, again in prompting and reinforcement, but should be used sparingly and appropriately.

V. DATA HANDLING

The format of a program is important to the user and to the learning process, but how a program handles data is equally important to the instructor. Data can be loaded into a program in three ways: string statements, data statements, and text files. The first two are typed directly into the program, making the programming simpler and offering some advantages in the random-versus-sequential option. But a knowledgeable user can LIST the program, and the drill or test answers will appear on the screen. Text files -- pockets of data -- can be reached only through the program. Secondly, changing string or data statements involves retyping part of the program, an always tedious and often impossible task. Text files offer easy changeability, as many of

the new authoring systems demonstrate. Lastly, a program should handle data output (dates, test numbers, scores, etc.) as well, storing it for future reference.

VI. REINFORCEMENT

One of the real advantages of CAI is immediate reinforcement for the learner, so this aspect should be examined closely in any program. A positive response should be rewarded and encouraged. A negative response should not just be judged as such, but the user should be shown the correct response and offered an explanation or recourse to an instructional frame. All reinforcements should have some variety beyond "Yes" and "No," and at some later stage in the program, the reinforcements might be removed. The user should be aware of his partial or interim score and shown his final score. Scores, questions, and other results should be able to be printed out, to carry reinforcement beyond the immediate session.

VII. PURPOSE

As noted in the introduction, software in English is proliferating, as is its use in writing labs. As CAI continues to grow, the purpose of both the CAI and a particular program should be clear to the user, the instructor, and the programmer (many English programs are written by computer science experts, who are unaware of English instructional needs). For example, a program might be designed to work with more than one user. Also, concepts presented in a program should be evaluated in a final test. Thirdly, programs are most effective when they are keyed to classroom texts or lab manuals, and when support material like handouts, printouts, or quizzes is available. Lastly, English

instructors should remember that CAI is a means, not an end, a process, not a product, and that English software should always suggest transference back to reading and writing, which are the real modes of instruction in college English.

COURSEWARE EVALUATION FORM Appendix A

PROGRAM_____ SUBJECT AREA_____ GRADE LEVEL_____

CRITERIA:

1. INSTRUCTIONS:

____ Are instruction presented clearly? Are instruction readable?
____ Are graphics or highlighting used?
____ Are examples given in instructions?
____ Is the purpose of the exercise clear to the reader?

2. USER CONTROL:

____ Are there disk-handling problems?
____ Does the program require used input (name, date, etc.)?
____ Can the user select the material (no. of questions, presentation)?
____ Does user control screen paging?
____ Are error users foreseen and controlled?
____ Can the user get help, re-enter, or exit at any point?
____ How much typing expertise is required?

3. SCREEN FORMAT:

____ Is the program on a 40 or 80 character screen?
____ Is the program all capitals letters, or normally mixed?
____ Are highlighting techniques (flashing or inverse words) used?
____ Are directions always present or available?
____ Are outlining or underlining techniques used?
____ Is each screen pleasantly and clearly spaced?

4. GRAPHICS:

____ Is color used, for gaining attention, or in scoring?
____ Do graphics have visual appeal?
____ Are graphics useful? just interesting? distractive?
____ Is sound used? appropriately?

5. DATA HANDLING:

____ How are data loaded into the program (strings, statements, files)?
____ Can data be easily changed?
____ Can data be input into a program (name, date, scores)?

6. REINFORCEMENT:

____ How does negative response guide learning?
____ How does positive response promote reinforcement?
____ Is interim scoring offered? final scoring?
____ Does scoring use graphics?
____ Can tests and/or results be printed?

7. PURPOSE:

____ Is the purpose clear to user, instructor, programmer?
____ What size group can use the program?
____ Are concepts evaluated in a final test?
____ Is the program keyed to a classroom text or lab manual?
____ Are support materials (handouts, quizzes, etc.) available?
____ Does the program suggest transference for its concepts?

REFERENCES: SOFTWARE EVALUATION

Bishop, Robert, et al. "Adapting Computer-assisted-instruction to the Non-Programmer." In RESOURCES IN EDUCATION. New York: Jan. 1974.

California Library Media Consortium. "Classroom Evaluation of Microcomputer Courseware." 1983.

Covington, Michael. "Some Trends in Computer-Aided Instruction that Ought Not to be Continued." Conference on College Composition and Communication, 1984.

Dilligan, Robert. "Computer Literacy: A Prerequisite for Evaluating CIA." Conference on College Composition and Communication, 1984.

Gillespie, Frank. "Instructional Design for Computer Assisted Instruction." In ACCESS (OCIS publication). Athens: Univ. of Georgia, Fall 1982.

Glass, R.L. SOFTWARE RELIABILITY GUIDEBOOK. Englewood, NJ: Prentice-Hall, 1979.

Heinrich, Mollenda, and Russell. "Appraisal Checklist: Computer-Assisted Instruction." In INSTRUCTIONAL MEDIA AND THE NEW TECHNOLOGIES OF INSTRUCTION. New York: Wiley and Sons, 1982.

Jaycox, Kathleen M. "Computer Applications in the Teaching of English." In RESOURCES IN EDUCATION. New York: July 1983.

Jordan, Pauline et al. "Community College English Lesson Index." In RESOURCES IN EDUCATION. Washington, D.C.: Sept. 1976.

Kenney, Sue. "Creating Lessons -- The Easier Way." In ACCESS. Athens: University of Georgia: Fall 1982.

Lump, Jane Marie. "Curiouser and Curiouser: Adventures in Softwareland." Conference on College Composition and Communication, 1984.

Miller, E. and W.E. Howden. TUTORIAL: SOFTWARE TESTING AND VALIDATION TECHNIQUES. IEEE-CS: 2nd edition, 1981.

Minnesota Educational Computing Consortium. "Computer Courseware Review." 1981.

Myers, G.C. THE ART OF SOFTWARE TESTING. New York: John Wiley & Sons, 1979.

Peters, L.J. SOFTWARE DESIGN: METHODS AND TECHNIQUES. Boston: Yourdon Press, 1981.

Whitworth College. "Computer-Based Instruction: A Pedigogical Approach." In RESOURCES IN EDUCATION. New York: June 1976.